 Oren Shriki, Nir Friedman, Erez Wolfson

Exercise #4 – PCA and EigenFaces

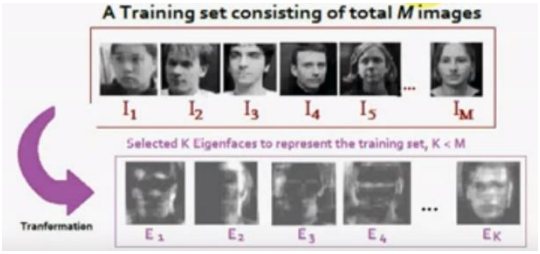
Background

- PCA is a popular method for dimensionality reduction and data compression.
- The goal of this exercise is to get familiar with PCA through analyzing a data-set of images of faces.

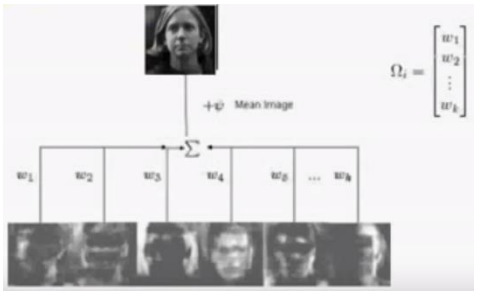
EigenFaces

1. Answer the following questions:
   a. Why would we want to lower the dimension of our data set?
   b. What are the disadvantages of reducing dimensionality?
   c. What are eigenvectors? What are eigenvalues? What does the size of the eigenvalues represent when using PCA?

2. EigenFaces: In this part you are going to perform PCA on a set of faces, compress them (with different parameters) and then reconstruct them. The principal components are called "EigenFaces".



After creating the EigenFaces, each face from the original dataset can be reconstructed using a linear combination of them:



3. Open 'ex4_part1.m' and run it.
The 'FacesAndEmotions.mat' data file will be loaded. It contains ***Images*** of faces with different emotions. The size of each image is *N* = Height*Width, *P* is the number of Images.

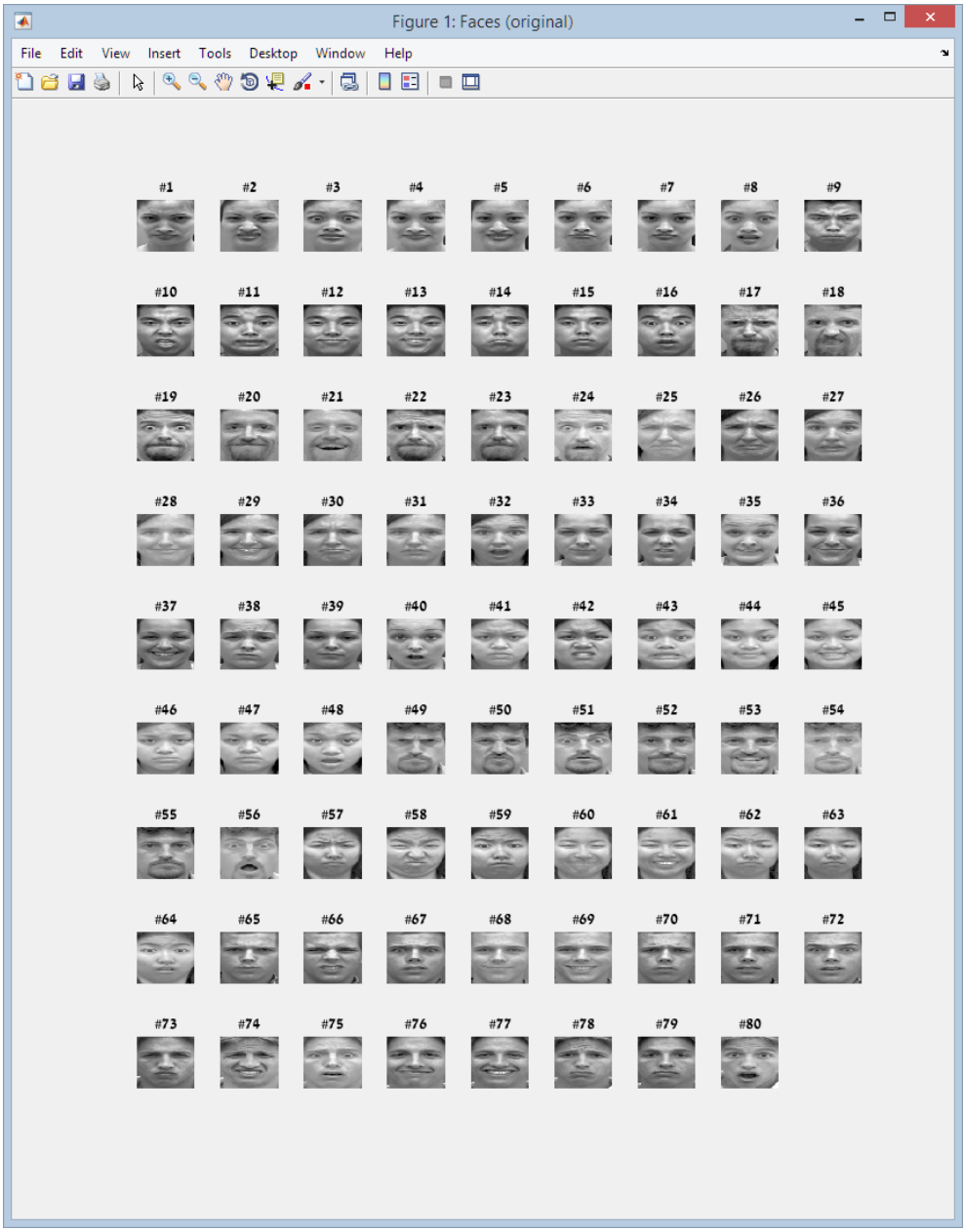
Neural data-analysis Workshop

Oren Shriki, Nir Friedman, Erez Wolfson

Follow the instructions in each section and implement the appropriate code.

Section #1 (*Plot original images*) should generate a figure like the following figure:

- The figure title should be 'Faces (original)' and images subplots should be titled with their index in the original Images structure.
- Use **`imagesc()`** and then **`colormap(gray)`** to plot a certain image according to its pixels intensities,.



Section #2 (*Perform PCA*):

- Vectorize the *Images* matrix before calculating the covariance matrix (you should reshape 'Images' *H*x*W*x*P* into *N*x*P*) and then **subtract the average face**. Use the following code to estimate the covariance matrix:

```
denom = P-1;
xCov = (X*X')./denom;
```

  Where **X** is the data matrix **after reshaping from 3D to 2D and subtracting the mean.**

- Alternatively, use the command **cov** (but read the help of the command and verify whether you should put **X** or **X'** as input).
- Calculate the Eigen Vectors and Eigen Values of the covariance matrix. Use the commands **eigs(xCov,P)**, **diag**. **DO NOT use any PCA related commands of Matlab.**

- Sort the eigenvectors according to the descending order of the corresponding eigenvalues.

Section #3 (*Plot Average face & EigenFaces*):

- Create a figure titled 'EigenFaces'.
- Show average face as 1st subplot (its title should be 'Average face').
- Iterate over all EFs/PCs and subplot them.
  Each EF title should be their corresponding "variance" (corresponding eigenvalue), formatted with 2 digits to the right of the decimal point.
- Think of the EFs order according to the PCA, before plotting them.
- You should have 81 images in this plot (1 Average face and the 80 eigenfaces with the largest eigenvalues).

Sections #4 (*Filter the variance*) & #5 (*eigenFaces scores*):

- Say you have a matrix V that contains the first M eigenvectors as columns. Compressing the data set using M principal components is equivalent to performing

$$y = V' * X \ (X \text{ is the matrix after subtracting the average face})$$

- The energy (variance) that is kept in the reconstruction (out of the total energy) is

$$\text{var} = \frac{\sum_{k=1}^{M} \lambda_k}{\sum_{k=1}^{N} \lambda_k} * 100 \ [\%]$$

  where λ_k is the k'th largest eigenvalue.

- Implement 4 cases, each case should be executed separately and have a different 'reconstruction' figure:

i. Keep all energy and reconstruct.

ii. Keep only the 1st PC and reconstruct (*Notice how much energy is stored in the first PC).*

iii. Keep 95% of the energy and reconstruct (*Notice how many components you were using*).

iv. Keep 80% of the energy and reconstruct (*Notice how many components you were using*).

- Use **`switch.. case`** and switching flag to choose which case to run.

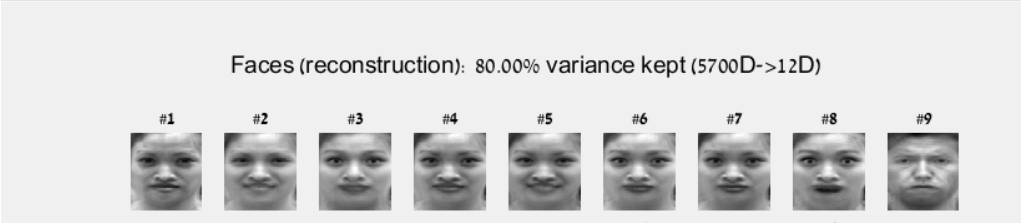
Sections #6 & #7 (*reconstruction*):

Reconstruct and plot it in a figure/s

- Reconstructing the original data using the compressed data is equivalent to performing

$$X_{estimated} = V * y + AvgFace$$

- The figures generated in this section should be titled
  "Faces (reconstruction): [energy kept] (ND->MD)".
  Images subplots should be titled with their index in the original Images structure.
  Here is an example:



Did you notice some brand new faces? How did they emerge? Think about it.

General notes:

- Remember to reshape before plotting an image (from 1D vector to 2D).
- For string formatting you can use **`sprintf()`** or simply concatenate strings using the **`num2str`**.

Exercise deliverables

- Submit according to the submission guidelines in Moodle.
- You will hand in a .zip file (NOT .rar or any other compression file type), containing:
  - MATLAB code main script.
  - Any additional functions you wrote.
  - A report containing at least:
    - Brief discussion of what you have done.
    - Answer all questions mentioned above (in blue).
    - Describe **briefly (No more than 3 sentences on each graph)** what you have done and accompany your results with discussion.
    - Include all figures as requested above - you should have 6 figures:
      - 1 figure with original faces.
      - 1 figure with average and 'Eigen Faces'.
      - 4 figures with reconstructed faces (1 for each case).

Good Luck!!